**Fonksiyon ve pointer kullanarak dikdörtgenin alanı ve cevresini hesaplama**

#include<stdio.h>

void hesapla(int \*p,int\*q,int \*n,int \*m)

{

\*n=\*p\*\*q;

\*m=2\*(\*p+\*q);

}

int main()

{

int \*a,\*b,alan,cevre;

printf("kısa,uzun=\n");

scanf("%d %d",&a,&b);

hesapla(&a,&b,&alan,&cevre);

printf("alan=,cevre=, %d,%d\n",alan,cevre);

return 0;

}

**Pointer kullanarak Dizinin ortalaması:**

#include<stdio.h>

#include<stdlib.h>

double ortalama(double dizi[],int n);

int main()

{

double a[4]={4,4,4,4};

double o;

o=ortalama(a,4);

printf("diznin ortalaması=%f\n",o);

return 0;

}

double ortalama(double dizi[],int n)

{

double \*p,t=0;

int i;

p=dizi;

for (i=0;i<n;i++)

{

t=t+\*(p+i);

}

return t/n;

}

**Pointerleri parametre olarak kullanmak:**

#include <stdio.h>

#include <stdlib.h>

void f1(int n)

{

n=66;

printf("f1 icinde : %d\n",n);

}

void f2(int \*n)

{

\*n=77;

printf("f2 icinde : %d\n",\*n);

}

int main()

{

int x=55;

printf("x'in degeri,\n");

printf("fonksiyonlar cagirilmadan once %d\n",x);

f1(x);

printf("f1 den sonra :%d\n",x);

f2(&x);

printf("f2 den sonra : %d\n",x);

return 0;

}

**Pointer kullanarak kopyalama :**

#include <stdio.h>

#include <stdlib.h>

int main()

{

char \*kelime1="merhaba dunya";

char \*kelime2;

char \*temp;

int baslangic\_index=3;

int kackarakter=8;

int i,j=0;

temp =&kelime1[baslangic\_index];

for (temp;temp<&kelime1[baslangic\_index+kackarakter-1];temp++)

{

kelime2[j]=\*temp;

j++;

}

kelime2[j]=NULL;

printf("kelime2:%s\n",kelime2);

return 0;

}

**Pointer kullanarak string ters cevirme**

#include<stdio.h>

int main()

{

char string[30];

printf("bir string giriniz:\n");

gets(string);

char \*being,\*end,temp;

being=string;

end=string;

while(\*end!=NULL)end++;

end--;

while (being<end)

{

temp=\*end;

\*end=\*being;

\*being=temp;

being++;

end--;

}

puts(string);

return 0;

}

**Pointer kullanarak binariyi decimala çevirme**

#include <stdio.h>

#include <math.h>

double binary2decimal(char \*ptr)

{

double deger = 0;

double indekssayisi = 0;

char \*begin, \*end;

begin = end = ptr;

while(\*end){ end++ ; indekssayisi++;} indekssayisi--;

while(begin < end)

{

if(\*begin == '1')

{

deger += pow(16, indekssayisi);

}

begin++;

indekssayisi--;

}

return deger;

}

int main()

{

char binary[40];

printf("Enter a binary: \n");

scanf("%s",binary); // gets(binary);

printf("Decimal of entered binary: %.0f\n",binary2decimal(binary));

return 0;

}

**Pointer kullanarak string aramak:**

#include<stdio.h>

int main()

{

char \*aranan="ara";

char \*aranacak="aranacak kelimeyi ara ara ";

char \*aranacakindex=aranacak,\*arananindex,\*temparama;

int sayac=0,kontrol;

while(\*aranacakindex)

{

arananindex=aranan;

kontrol=0;

if (\*aranacakindex==\*arananindex)

{

arananindex++;

temparama=aranacakindex+1;

while(\*arananindex&&\*temparama)

{

if (\*temparama!=\*arananindex)

{

kontrol=1;

break;

}

arananindex++;

temparama++;

}

if(kontrol==0&&!\*arananindex)

{

sayac++;

}

}

aranacakindex++;

}

printf("sayac:%d",sayac);

return 0;

}

**Pointer kullanarak bir string içerisindeki sesli harfleri silme:**

#include <stdio.h>

#include <stdlib.h>

int main()

{

char aranacak[30];

printf("string giriniz:\n");

gets(aranacak);

char \*aranacakindex=aranacak;

while(\*aranacakindex)

{

if(\*aranacakindex=='a'||\*aranacakindex=='o'||\*aranacakindex=='e'||\*aranacakindex=='i'||\*aranacakindex=='ö'||\*aranacakindex=='u'||\*aranacakindex=='ı'||\*aranacakindex=='ü')

{

}

else printf("%c",\*aranacakindex);

\*aranacakindex++;

}

return 0;

}

**Pointer kullanarak string içindeki karakterleri sıralamak:**

#include <stdio.h>

#include <stdlib.h>

int main()

{

char aranacak[30];

printf("string giriniz:\n");

gets(aranacak);

char \*temp;

int uzunluk = 0;

int m = 0,n = 0,sayac;

char \*aranacakindex=aranacak;

uzunluk = strlen(aranacak);

for( m = 0 ; m < uzunluk - 1; m++ )

{

for( n = m+1 ; n < uzunluk; n++ )

{

if( \*(aranacakindex+m) > \*(aranacakindex+n) )

{

temp = \*(aranacakindex+m);

\*(aranacakindex+m) = \*(aranacakindex+n);

\*(aranacakindex+n) = temp;

}

}

}

for(int j=0;j<uzunluk;j++)printf("%c",\*(aranacakindex+j));

return 0;

}

**Pointer kullanarak iki matrisin çarpımı:**

#include <stdio.h>

#include <stdlib.h>

int main()

{

int a[3][2]={{2,3,5},{2,1},{1,2,4}};

int b[2][3]={{12,2,3},{5,2,2}};

int c[3][3]={0},i,j,k;

int \*adresc=c;

int \*adresb=b;

int \*adresa=a;

for(i=0;i<3;i++)

{

for(j=0;j<3;j++)

{

for(k=0;k<2;k++)

{

\*(adresc+(i\*3)+j)+=\*(adresa+(i\*2)+k)+\*(adresb+(k\*3)+j);

}

}

printf("\n");

}

printf("sonuc: \n");

for(i=0;i<3;i++)

{

adresc=c+i;

for(k=0;k<3;k++)

{

printf("\t%d",\*(adresc+k));

}

printf("\n");

}

return 0;

}

**Conts pointer:**

#include <stdio.h>

#include <stdlib.h>

int main()

{

int var1=1,var2=2;

int \*const ptr =&var1;

printf("ptr : %d\n",\*ptr);

//ptr=&var2; aktif yaparsak hata verir

\*ptr=11;

printf("ptr:%d\n",\*ptr);

return 0;

}

**Pointer to const:**

#include <stdio.h>

#include <stdlib.h>

int main()

{

int var1=1,var2=2;

const int \*ptr =&var1;

printf("ptr : %d\n",\*ptr);

//\*ptr=1; aktif yaparsak hata veriri

ptr=&var2;

printf("ptr:%d\n",\*ptr);

return 0;

}

**Const Pointer to const:**

#include <stdio.h>

#include <stdlib.h>

int main()

{

int var1=1,var2=2;

const int \* const ptr =&var1;

//\*ptr=1; aktif yaparsak hata veriri

// ptr=&var2;

printf("ptr:%d\n",\*ptr);

return 0;

}

**Fonksiyonu pointer olarak kullanıp çıkarma işlemi yapmak:**

#include <stdio.h>

#include <stdlib.h>

int subtract(int x,int y)

{

return x-y;

}

int main()

{

int (\*subtractptr)(int,int);

subtractptr=&subtract;

//int (\*subtractptr)(int,int)=&subtract; buda aynı

int z=subtractptr(10,2);

printf("subtract gives %d\n",z);

return 0;

}

**Pointer to pointer:**

#include <stdio.h>

#include <stdlib.h>

int main()

{

int x=5;

int \*p=&x;

int \*\*q=&p;

int \*\*\*r=&q;

printf("%d\n",p);//p nin adresi

printf("%d\n",\*p);//p nin degeri

printf("%d\n",q);//q nun adresi

printf("%d\n",\*q);//p nin adresi

printf("%d\n",\*\*q);//q nun degeri

printf("%d\n",r);//r nin adresi

printf("%d\n",\*r);//q nun adresi

printf("%d\n",\*\*r);//p nin adresi

printf("%d\n",\*\*\*r);//r nin degeri

\*p=6;

printf("x=%d\n",x);

\*\*q=7;

printf("x=%d\n",x);

\*\*\*r=10;

printf("x=%d\n",x);

return 0;

}

**Malloc,Calloc,Realloc:**

**Malloc:** istenen değer kadar bellekten yer ayırır ve değerleri bellekte o an ne varsa odur.

**Calloc:** istenen değer kadar bellekten yer ayırır ve değerleri hepsi sıfırdır.

**Realloc:** bellekte yer azaltma veya çoğaltma için kullanılır.

**Örnek 1:**

#include<stdio.h>

#include<stdlib.h>

int main()

{

int \*dizi;

int i,n;

dizi=(int \*)malloc(sizeof(int)\*1);

printf("Degisken degerini gir: ");

scanf("%d",dizi);

printf("Dizi degisken adresi :%p, deger: %d",dizi,\*dizi);

return 0;

}

**Örnek 2:**

#include<stdio.h>

#include<stdlib.h>

int main()

{

int \*dizi;

int i,n;

printf("\n\n");

printf("Dizi boyutu gir: ");

scanf("%d",&n);

dizi=(int \*)malloc(n\*sizeof(int));

printf("Malloc adres: %p, dizi boyutu: %d ",dizi, \_msize(dizi));//\_msize()dizinin boyutunu döndürür

for(i=0;i<n;i++)

{

printf("%d.eleman degerini gir: ",i+1);

scanf("%d",dizi+i); //i=i\*sizeof(int)

}

free(dizi);

dizi=NULL;

printf("\n");

printf("Free adres: %p,dizi boyut: %d",dizi,\_msize(dizi));

return 0;

}

**Örnek 3:**

#include<stdio.h>

#include<stdlib.h>

int main()

{

int \*dizi;

int i,n;

printf("Dizi boyutunu girin: ");

scanf("%d",&n);

dizi=(int \*)calloc(n,sizeof(int)); //verdiği bellek boş geliyor, mallocta dolu

printf("Calloc adres: %p, dizi boyut: %d\n",dizi,\_msize(dizi));

for(i=0;i<n;i++)

{

printf("%d .eleman degeri: %d\n",i+1,dizi[i]); //\*(dizi+i)

}

dizi=(int \*)realloc(dizi,n\*sizeof(int)\*2000);

printf("Calloc adres: %p, dizi boyut: %d\n",dizi,\_msize(dizi));

dizi=(int \*)realloc(dizi,0);

printf("Realloc adres: %p, dizi boyut: %d\n",dizi,\_msize(dizi));

return 0;

}

**Örnek 4:**

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

int main()

{

char \*buffer;

buffer=(char \*)malloc(10);//sizeof yazmaya gerek yok zaten 1

if(buffer==NULL)

{

printf("Malloc failed!");

exit(1);//programdan çıkmak için,ondan sonraki kodları çalıştırmaz

}

printf("Buffer size: %d\n",\_msize(buffer));

strcpy(buffer,"Test Ok"); //iki parametre alır birinci hedef ikincisi kaynak ikinic degeri 1.degere aktarır yeterli alan varsa

printf("Buffer contains: %s\n",buffer);

if((buffer=(char \*)realloc(buffer,5))==NULL)

{

printf("Realloc failed!");

exit(1);

}

printf("Buffer of size after: %d\n",\_msize(buffer));

strcpy(buffer,"Test");

printf("Buffer now contains: %s\n",buffer);

free(buffer);

return 0;

}

**Örnek 5:**

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

char \*altKatar(char \*str,int,int);

char \*alt;

int main()

{

int i;

char \*s, \*parca;

s="programlama";

for(i=0;i<strlen(s);i++)

{

parca=altKatar(s,0,i);

printf("Adres: %d deger: %s \n ",parca,parca); //\t tab kadar bosluk býrakr

free(parca);

}

return 0;

}

char \*altKatar(char \*str,int p1,int p2)

{

int i,j=0,n;

n=p2-p1;

alt=(char \*)calloc(n+1,sizeof(char));

for(i=p1;i<=p2;i++)

{

alt[j++]=str[i];

}

alt[j]='\0';

return alt;

}

**Örnek 6:**

#include<stdio.h>

#include<stdlib.h>

int main()

{

int \*\*matris;

int satir,kolon,i,k,j,s;

printf("Matrisin satir sayisi: ");

scanf("%d",&satir);

printf("Matrisin sutun sayisi: ");

scanf("%d",&kolon);

matris=(int \*\*)calloc(satir,sizeof(int));

for(i=0;i<satir;i++)

{

matris[i]=(int \*)calloc(kolon,sizeof(int));

}

for(i=0;i<satir;i++)

{

for(j=0;j<kolon;j++)

{

printf("Matris elemanini girin -> matris[%d][%d]: ",i,j);

scanf("%d",&(matris[i][j]));

}

}

printf("\n");

for(i=0;i<satir;i++)

{

for(j=0;j<kolon;j++)

{

printf("%-4d",matris[i][j]);//%4d= 4 karakterlik yer ayırır//başına - yazınca sola dayalı yazıyor

}

printf("\n");

}

for(i=0;i<satir;i++)

{

free((void\*)matris[i]);

}

free((void\*)matris);

return 0;

}

**Struct yapılar:**

**Örnek1:**

#include <stdio.h>

#include <stdlib.h>

struct kayit

{

char ad[10];

char \*soyad;

long no;

int sinif;

};

int main()

{

struct kayit ogrenci;

printf("ogrenci numarasi:");

scanf("%ld",&ogrenci.no);

printf("ogrenci adi:");

scanf("%s",&ogrenci.ad);

printf("ogrenci soyadi:");

scanf("%s",&ogrenci.soyad);

printf("ogrenci sinif:");

scanf("%d",&ogrenci.sinif);

printf("ogrenci bilgileri:");

printf("ogrenci numarasi:%ld\n",ogrenci.no);

printf("ogrenci adi:%s\n",&ogrenci.ad);

printf("ogrenci soyadi:%s\n",&ogrenci.soyad);

printf("ogrenci sinif:%d\n",ogrenci.sinif);

return 0;

}

**Örnek 2:**

#include <stdio.h>

#include <stdlib.h>

struct dogumgunu

{

int yil;

int ay;

int gun;

};

struct dogumgunu dogum\_gunu,bugun,fark;

int main()

{

printf("dogum gununuzu gg-aa-yyyy giriniz:");

scanf("%d-%d-%d",&dogum\_gunu.gun,&dogum\_gunu.ay,&dogum\_gunu.yil);

printf("bugun tarihi gg-aa-yyyy giriniz:");

scanf("%d-%d-%d",&bugun.gun,&bugun.ay,&bugun.yil);

fark.gun=bugun.gun-dogum\_gunu.gun;

if(fark.gun<0)

{

fark.gun+=30; //fark.gun=fark.gun+30;

bugun.ay--;

}

fark.ay=bugun.ay-dogum\_gunu.ay;

if(fark.ay<0)

{

fark.ay+=12; //fark.gun=fark.gun+30;

bugun.yil--;

}

fark.yil=bugun.yil-dogum\_gunu.yil;

printf("\nfark: %d/%d/%d\n",fark.gun,fark.ay,fark.yil);

return 0;

}

**Örnek 3:**

#include <stdio.h>

#include <stdlib.h>

struct dogum

{

int yil,ay,gun;

};

struct ogrenci

{

char ad[20];

int boy;

struct dogum dogumbilgileri;

int sinif;

}kisi={"mehmet",178,{1980,11,24},2};

int main()

{

printf("ad: %s\n",kisi.ad);

printf("dogum bilgileri: %d/%d/%d\n",kisi.dogumbilgileri.gun,kisi.dogumbilgileri.ay,kisi.dogumbilgileri.yil);

return 0;

}

**Örnek 4:**

#include <stdio.h>

#include <stdlib.h>

#define n 3

typedef struct ders

{

int derskodu;

float vizenotu;

float finalnotu;

}Ders;

int main()

{

Ders dersler[n];

int i;

for(i=0;i<n;i++)

{

printf("%d. ders kodu :",i+1);

scanf("%d",&dersler[i].derskodu);

printf("%d. dersin vizesi :",i+1);

scanf("%f",&dersler[i].vizenotu);

printf("%d. dersin finali:",i+1);

scanf("%f",&dersler[i].finalnotu);

}

for(i=0;i<n;i++)

{

printf("%d\t%.2f\t%.2f\t%.2f\n",dersler[i].derskodu,

dersler[i].vizenotu,

dersler[i].finalnotu,

0.4\*dersler[i].vizenotu+0.6\*dersler[i].finalnotu);

}

return 0;

}

**Örnek 5:**

#include <stdio.h>

#include <stdlib.h>

union foo

{

int a;

char b;

};

int main()

{

union foo x;

x.a=15;

printf("x.a=%d\n",x.a);

printf("x.b=%c\n",x.b);

x.b='c';

printf("x.a=%d\n",x.a);

printf("x.b=%c\n",x.b);

return 0;

}

**Örnek 6:**

#include <stdio.h>

#include <stdlib.h>

struct name

{

int a;

float b;

};

int main()

{

struct name p,\*ptr;

ptr=&p;

printf("integer value:");

scanf("%d",&(\*ptr).a);

printf("float value:");

scanf("%f",&ptr->b);

printf("\n--------\n");

printf("%d\t%.2f",(\*ptr).a,ptr->b);

return 0;

}

**Örnek 7:**

#include <stdio.h>

#include <stdlib.h>

struct name

{

int model;

float kilometre;

char marka[30];

};

int main()

{

struct name \*ptr;

int n,i;

printf("kac araba kaydedilecek:");

scanf("%d",&n);

ptr= (struct name\*)malloc(n\*sizeof(struct name));

for (i=0;i<n;i++)

{

printf("araba bosluk olacak sekilde marka, model, ve km degerlerini giriniz ");

scanf("%s %d %f",(ptr+i)->marka,&(ptr+i)->model,&(ptr+i)->kilometre);

}

printf("\n----------\n");

for (i=0;i<n;i++)

{

printf("%-12s %-4d %-.2f\n",(ptr+i)->marka,(ptr+i)->model,(ptr+i)->kilometre);

}

return 0;

}